Package: dyadMLM (via r-universe)

October 28, 2026

Title Tools for Dyadic Multilevel Models

Version 0.3.0

Date 2026-10-27

Description Provides tools for dyadic multilevel modeling with linear and generalized linear mixed-effects models. It validates and prepares long-format cross-sectional and intensive longitudinal data, including ecological momentary assessment designs, for distinguishable and exchangeable dyads. It also supports datasets containing multiple observed dyad compositions. It constructs composition-aware, model-ready variables for Actor-Partner Interdependence Models (APIMs), Dyad-Individual Models (DIMs), and Dyadic Score Models (DSMs). Prepared data can be used with model engines such as 'glmmTMB' and 'brms' for Gaussian and non-Gaussian outcomes, including counts, proportions, and skewed continuous responses. Post-estimation tools compare compatible fitted models and back-transform exchangeable sum-and-difference random-effect covariance structures into member-level quantities. Experimental predictive checks compare observed and simulated response variances and partner correlations in Gaussian and selected non-Gaussian models. Distributional diagnostics separately check response distributions and residual patterns, using simulations that retain partner and time dependence. The APIM and DSM specifications and their relationships follow Iida et al. (2018) <doi:10.1177/0265407517725407>; the multilevel sum-and-difference random-effects implementation for exchangeable dyads adapts del Rosario and West (2025) <doi:10.1177/25152459251351286>.

License GPL-3

Encoding UTF-8

RoxygenNote 8.0.0

Suggests brms, glmmTMB (>= 1.1.11), htmltools, knitr, lavaan, lifecycle, marginaleffects, qgam, rmarkdown, spelling, testthat (>= 3.1.7)

Config/testthat/edition 3

Imports cli, dplyr (>= 1.1.1), pillar, rlang (>= 0.4.0), tidyselect, tibble (>= 3.0.0), withr (>= 3.0.3)

VignetteBuilder knitr, rmarkdown

Depends R (>= 4.1)

LazyData true

URL https://pascal-kueng.github.io/dyadMLM/, https://github.com/Pascal-Kueng/dyadMLM

BugReports https://github.com/Pascal-Kueng/dyadMLM/issues

Language en-US

NeedsCompilation no

Author Pascal Küng [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-7346-9414>)

Maintainer Pascal Küng <kueng.pascal@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-28 00:20:02 UTC

RemoteUrl https://github.com/cran/dyadMLM

RemoteRef HEAD

RemoteSha 6a44921b9adc786f120413beecb226249120488a

Contents

check_dyad_outcomes 3
check_dyad_residuals 5
check_partner_dependence 8
compare_nested_models 13
dyads_cross 15
dyads_ild 16
dyads_nbinom_cross 17
dyads_nbinom_ild 18
prepare_dyad_data 19
recover_exchangeable_covariance 23
simulate_dyad_responses 27

Index **30**

check_dyad_outcomes *Check outcome distributions against model simulations*

Description

[Experimental] A model should reproduce the distribution of the observed outcomes. This check compares the observed outcome distribution and summaries (response SD after subtracting predictions, largest absolute deviation from prediction, number of zeros) with complete datasets from `simulate_dyad_responses()`. This helps identify mismatches in the model's assumptions, such as the distribution family, dispersion, or zero inflation. For PIT residual patterns, also run `check_dyad_residuals()` with the same simulations. For guidance on how to read the plots, see `plot.dyadMLM_outcome_check()`.

Usage

```
check_dyad_outcomes(
  simulations,
  dyad = NULL,
  role = NULL,
  member = NULL,
  plot = TRUE,
  ask = NULL,
  panels = TRUE,
  data = NULL
)
```

Arguments

`simulations`	An object from `simulate_dyad_responses()`. At least 1,000 simulated datasets are recommended for stable comparisons.
`dyad`	Dyad ID column, with or without quotes. Required when `role` is supplied.
`role`	Role column, with or without quotes, such as gender. `NULL` (default) pools all observations. Otherwise, each composition (pair of partners' roles, such as female-male) is checked separately. Each member must keep the same role. With few dyads in a composition, the simulated ranges are wide, so its panels lack sensitivity.
`member`	Person ID column, with or without quotes. Required with `role` when dyads have repeated observations.
`plot`	If `TRUE` (default), draw the comparison plots for visual checks. If `FALSE`, `ask` and `panels` are ignored.
`ask`	Whether to pause between figures on an interactive device. `NULL` (default) pauses when there is more than one figure, `TRUE` pauses and `FALSE` draws without pausing. File devices never pause.
`panels`	If `TRUE` (default), arrange checks in figures by composition, with one column per role or a pooled column. If `FALSE`, draw each check and role separately.

`data` The unchanged data used to fit the model. Supply it when the columns you name (such as `role`) are not in the model formulas. Also supply it when rows were excluded during fitting (e.g., for missing values). Otherwise, a person whose partner's rows were excluded has an unknown composition and is omitted.

Value

Invisibly returns a `dyadMLM_outcome_check` list with role-specific summary matrices and outcome distributions in `compositions`. Matrices have one column per dataset (observed first, then all simulated). Optionally, suppress plotting with `plot = FALSE` and draw the saved result later with `plot(result)`, which also takes `ask` and `panels`.

Scope

Complete simulated datasets keep the fitted partner and time dependence and any modeled differences in variability between roles. Unlike DHARMa's default (since version 0.5.0), simulations redraw all random effects. Dyadic models often fix the residual variance near zero (`dispformula = ~ 0`) and model the partners' residual variances and covariance as random effects. If held fixed, these random effects would reproduce the data almost exactly. Parameters stay fixed at their fitted values, so parameter uncertainty is not included. These are descriptive checks, not significance tests, and differ from DHARMa's plots and tests (see the distribution-checks article). They do not directly assess serial correlation.

Predicted outcomes are mean predictions with random effects set to zero (see `simulate_dyad_responses()`). The checks use the same fitted rows and families as `simulate_dyad_responses()` and weight all observations equally. Check partner correlations with `check_partner_dependence()`.

See Also

`check_dyad_residuals()`, `check_partner_dependence()`

Examples

```
example_data <- prepare_dyad_data(
  dyads_cross,
  dyad = coupleID,
  member = personID,
  model_types = "none",
  seed = 123
)
# Exchangeable dyads: the two random-effect blocks represent the partners'
# residual variances and covariance.
model <- glmmTMB::glmmTMB(
  closeness ~ gender + provided_support +
    us(1 | coupleID) + us(0 + .member_contrast_arbitrary | coupleID),
  dispformula = ~ 0,
  data = example_data
)
simulations <- simulate_dyad_responses(
  model,
  nsim = 100, # Low for a quick example. Use at least 1,000.
  seed = 123
```

```
)
check_dyad_outcomes(
  simulations,
  dyad = coupleID,
  role = gender,
  ask = FALSE
)
```

`check_dyad_residuals` *Check residual distributions and patterns*

Description

[Experimental] A model should reproduce how responses are distributed around its predictions. This check finds where each observed outcome falls among its simulated values (PIT residuals), using complete datasets from `simulate_dyad_responses()`. This helps identify mismatches in the model's assumptions, such as the distribution family, dispersion, or predictor effects. For checks on the outcome scale, also run `check_dyad_outcomes()` with the same simulations. For guidance on how to read the plots, see `plot.dyadMLM_residual_check()`.

Usage

```
check_dyad_residuals(
  simulations,
  dyad = NULL,
  role = NULL,
  member = NULL,
  predictors = NULL,
  seed = 123,
  plot = TRUE,
  ask = NULL,
  panels = TRUE,
  data = NULL,
  quartile_lines = c("bins", "qgam")
)
```

Arguments

`simulations`	An object from `simulate_dyad_responses()` with at least 200 simulated datasets. At least 1,000 are recommended for stable comparisons.
`dyad`	Dyad ID column, with or without quotes. Required when `role` is supplied.
`role`	Role column, with or without quotes, such as gender. `NULL` (default) pools all observations. Otherwise, each composition (pair of partners' roles, such as female-male) is checked separately. Each member must keep the same role. With few dyads in a composition, the simulated ranges are wide, so its panels lack sensitivity.

member	Person ID column, with or without quotes. Required with role when dyads have repeated observations.
predictors	Optional quoted column names, e.g. c("age", "stress"). Each adds a page of PIT quartiles across that column, to show effects the model omits or may misspecify, especially partner effects. Missing or infinite values are left out of that page only. NULL (default) adds no predictor pages.
seed	Seed for randomized PIT residuals. The caller's random-number state is restored afterwards. With NULL, use and advance the caller's random-number state.
plot	If TRUE (default), draw the comparison plots for visual checks. If FALSE, ask and panels are ignored.
ask	Whether to pause between figures on an interactive device. NULL (default) pauses when there is more than one figure, TRUE pauses and FALSE draws without pausing. File devices never pause.
panels	If TRUE (default), arrange checks in figures by composition, with one column per role or a pooled column. If FALSE, draw each check and role separately.
data	The unchanged data used to fit the model. Supply it when the columns you name (such as role) are not in the model formulas. Also supply it when rows were excluded during fitting (e.g., for missing values). Otherwise, a person whose partner's rows were excluded has an unknown composition and is omitted.
quartile_lines	Quantile regression lines ("qgam") can show trends near the ends of a numeric axis more clearly than smoothed bins ("bins", default). As in DHARMa, they are fitted with the qgam package (Fasiolo et al., 2021) and drawn on ranks of the axis values. Each dataset gets its own lines. Tuning qgam for every dataset would take far too long, so simulated datasets reuse the tuning chosen for the observed data. The 95% coverage of the qgam envelopes is therefore approximate. "qgam" is much slower and may warn when fits do not converge.

Value

Invisibly returns a dyadMLM_residual_check list with the pit matrix (after any centring) and calculated summaries in compositions. pit has one row per fitted observation and one column per dataset (observed first, then all simulated). Optionally, suppress plotting with plot = FALSE and draw the saved result later with plot(result), which also takes ask and panels.

Scope

Complete simulated datasets keep the fitted partner and time dependence and any modeled differences in variability between roles. Unlike DHARMa's default (since version 0.5.0), simulations redraw all random effects. Dyadic models often fix the residual variance near zero (dispformula = ~ 0) and model the partners' residual variances and covariance as random effects. If held fixed, these random effects would reproduce the data almost exactly. Parameters stay fixed at their fitted values, so parameter uncertainty is not included. These are descriptive checks, not significance tests, and differ from DHARMa's plots and tests (see the distribution-checks article). They do not directly assess serial correlation.

Predicted outcomes are mean predictions with random effects set to zero (see simulate_dyad_responses()). The checks use the same fitted rows and families as simulate_dyad_responses() and weight all observations equally. Check partner correlations with check_partner_dependence().

Method and credit

The internal calculation follows the simulation-based PIT approach used in Florian Hartig's DHARMa package and the randomized quantile residual method of Dunn and Smyth (1996). We implement the definition directly and keep the uniform 0–1 scale.

At each fitted row, rank the observed and all simulated outcomes together, breaking ties randomly. For rank `r` among `m` datasets, use `(r - U) / m`, where `U` is uniform between 0 and 1.

When the model freely estimates an overall location (an intercept or separate role means) and is not ordinal, PIT is centred automatically: each dataset's PIT is transformed to normal scores, centred on its median across all fitted rows, and transformed back. Panels then show residuals relative to this overall location, so all panels can show less of an overall shift. Centring also partly offsets how fitting makes departures look smaller.

Global envelopes use extreme-rank-length ordering (Myllymäki et al., 2017).

References

Dunn, P. K., & Smyth, G. K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics, 5*(3), 236–244. doi:10.2307/1390802

Fasiolo, M., Wood, S. N., Zaffran, M., Nedellec, R., & Goude, Y. (2021). qgam: Bayesian nonparametric quantile regression modeling in R. *Journal of Statistical Software, 100*(9), 1–31. doi:10.18637/jss.v100.i09

Hartig, F. (2026). *DHARMa: Residual diagnostics for hierarchical (multi-level/mixed) regression models* (Version 0.5.0) [R package]. doi:10.32614/CRAN.package.DHARMa

Myllymäki, M., Mrkvička, T., Grabarnik, P., Seijo, H., & Hahn, U. (2017). Global envelope tests for spatial processes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), 79*(2), 381–404. doi:10.1111/rssb.12172

See Also

`check_dyad_outcomes()`, `check_partner_dependence()`

Examples

```
example_data <- prepare_dyad_data(
  dyads_cross,
  dyad = coupleID,
  member = personID,
  model_types = "none",
  seed = 123
)
# Exchangeable dyads: the two random-effect blocks represent the partners'
# residual variances and covariance.
model <- glmmTMB::glmmTMB(
  closeness ~ gender + provided_support +
    us(1 | coupleID) + us(0 + .member_contrast_arbitrary | coupleID),
  dispformula = ~ 0,
  data = example_data
)
simulations <- simulate_dyad_responses(
```

```
  model,
  nsim = 200, # Low for a quick example. Use at least 1,000.
  seed = 123
)
check_dyad_residuals(
  simulations,
  dyad = coupleID,
  role = gender,
  predictors = "provided_support",
  ask = FALSE
)
# Save a pooled check without drawing, then plot each figure separately.
result <- check_dyad_residuals(
  simulations,
  role = NULL,
  predictors = "provided_support",
  plot = FALSE
)
plot(result, panels = FALSE, ask = FALSE)
```

`check_partner_dependence` *Check whether a fitted model reproduces partner dependence*

Description

[Experimental] A dyadic model should reproduce how much responses vary and how strongly partners' responses are related. This check compares the observed response variances and partner correlations with those of responses simulated from the fitted model. This helps identify mismatches in the model's assumptions. This check is especially useful when a less restricted model does not converge and so cannot be used for model comparison. For repeated occasions, supply `member` and `time`.

Usage

```
check_partner_dependence(
  simulations,
  dyad,
  role = NULL,
  member = NULL,
  time = NULL,
  plot = TRUE,
  response = c("model-centred", "raw"),
  ask = NULL,
  panels = TRUE,
  data = NULL
)
```

Arguments

simulations	An object returned by simulate_dyad_responses().
dyad	The dyad column name. Looked up first in the fitted model frame, then in data if supplied.
role	The role column name. Looked up first in the fitted model frame, then in data if supplied. NULL (default) pools all dyads as exchangeable. Supply roles even for an exchangeable model to reveal variance or partner-correlation mismatches that pooling may hide. Each dyad composition (role pair, such as female-male) is checked separately, using exchangeable summaries for same-role pairs and role-specific summaries otherwise. For repeated occasions, each member must keep the same role within their dyad. Rows with missing roles are omitted.
member	The column name identifying each member. Only needed for repeated occasions, ignored without a time argument. Looked up first in the fitted model frame, then in data if supplied.
time	The column name identifying each occasion. Only needed for repeated occasions. Requires member to be supplied. time must identify occasions that partners exactly share, such as the diary day. Every fitted row needs a known member and time. Looked up first in the fitted model frame, then in data if supplied.
plot	If TRUE (default), draw the comparison plots for visual checks. If FALSE, ask and panels are ignored.
response	"model-centred" (default) subtracts the same model predictions from observed and simulated responses. The predictive check then assesses whether the model reproduces the variance and partner correlations remaining after accounting for its fixed-effect predictions. This includes random effects and observation-level noise. With "raw", the predictive check assesses whether the full model, including fixed effects, reproduces the overall response variances and partner correlations. See simulate_dyad_responses() for how predictions are defined.
ask	Whether to pause between figures on an interactive device. NULL (default) pauses when there is more than one figure, TRUE pauses and FALSE draws without pausing. File devices never pause.
panels	If TRUE (default), show each composition in one figure, with two rows and up to three columns. If FALSE, draw each statistic separately. Graphics settings are restored afterwards.
data	Optional: the unchanged data frame used to fit the model. Supply it when dyad, role, member, or time is not in the model formulas.

Value

The comparison plots (shown by default) are the main output. The function invisibly returns a dyadMLM_partner_check object containing the compositions table with pair counts and a statistics tibble for each composition. Compositions with fewer than three complete dyads are skipped and have NULL statistics. Each tibble has one observed row followed by one row per simulation, identified by dataset. summary compares each observed statistic with the middle 95% of its simulations (lower, upper) and marks those outside it. Rows with missing dyad IDs or roles and incomplete dyads are omitted with a warning. The object includes omission counts and settings,

and can be saved and plotted later. With repeated occasions, `compositions` has a `level` column (`"between"` or `"within"`), and each label ends with its level. Within-level rows count the dyads that share at least one occasion.

Reading the plots

Blue histograms show simulated summaries. Red lines mark observed values. Dashed lines enclose the middle 95% of simulations (no formal confidence intervals). The heading shows the dyad composition, its number of usable dyads, and the total across all compositions.

The top row compares:

- **Response SDs:** one for each role, or one common SD for exchangeable members.
- **Partner correlation:** how strongly partners' responses are related.

The bottom row shows the same information using dyad averages and half-differences:

- **Dyad-average SD:** how much dyads differ in their average response.
- **Half-difference SD or RMS:** each partner difference is divided by two. With distinct roles, the SD shows how much signed differences vary across dyads. For exchangeable members, the RMS (root mean square) shows their typical size, regardless of partner order.
- **Dyad-average/role-difference correlation:** shown only for distinct roles, because it depends on how partners are ordered. Positive values indicate greater variance for the first named role. Negative values indicate greater variance for the second.

An observed value far from most simulated values may indicate that the model does not reproduce that feature of the data well. Even if simulated and observed values agree, this does not show that omitted dependence is negligible, especially with few dyads. The partner-dependence study illustrates how sample size affects detection when residual partner correlation is omitted.

Checking each composition separately can reveal differences hidden by pooling. With few dyads in a composition, the simulated ranges are wide, so its panels lack sensitivity. Flags (observed values outside the middle 95%, marked `*` when printed) can occur by chance, especially when checking several summaries. They are a starting point for further investigation, not necessarily a reason to reject the model. The covariance-pooling study illustrates detection and false alarms when checking each composition.

A flexible covariance model will usually reproduce features it estimated from the same data. Agreement alone therefore does not establish good fit. For example, pooled summaries (`role = NULL`) of a Gaussian model with an unconstrained exchangeable covariance, as below, agree almost by construction. Use a suitable model comparison to formally test a specific covariance restriction when both models can be fitted (see `compare_nested_models()`).

Repeated occasions

With repeated occasions, partners can be related in two ways: in their mean responses, and in being above or below their own means on the same occasions. A model should reproduce both. With `member` and `time`, every dataset (observed and simulated) is split into two levels after any model-centring. Each level is checked like cross-sectional data:

- **Between:** each member's mean over all their fitted occasions, paired within the dyad. Each dyad counts once.

- **Within:** each response minus its member's mean, paired with the partner's deviation on the same occasion. Each shared occasion counts once.

An occasion without the partner still counts in the member's mean.

The within check is related to repeated-measures correlation (Bakdash & Marusich, 2017).

The longitudinal study examines detection and false alarms. With 50 or fewer dyads, the check often missed omitted partner covariance.

Known limitations and caveats:

- A misfit at one level can also impact the other level's summaries, especially with few occasions (due to member means including occasion-level variance) or nonlinear links. Thus, both levels should be interpreted together.
- Check dependence across occasions (e.g., autocorrelation) separately. If the model omits temporal dependence, its covariance estimates can be biased, and these checks may not reveal it.
- These checks can be misleading when the model includes lagged outcomes as predictors.

Technical details

After any centring, paired responses `a` and `b` are used to compute dyad averages `M = (a + b) / 2` and half-differences `D = (a - b) / 2`. Roles follow factor levels or sorted values.

For exchangeable members, common member variance is `var(M) + mean(D^2)` and partner covariance is `var(M) - mean(D^2)`. Partner correlation is covariance divided by variance. Half-difference RMS is `sqrt(mean(D^2))`.

The variance and covariance calculations for exchangeable dyads follow Woody and Sadler (2005).

References

Woody, E., & Sadler, P. (2005). Structural equation models for interchangeable dyads: Being the same makes a difference. *Psychological Methods, 10*(2), 139–158. doi:10.1037/1082989X.10.2.139

Bakdash, J. Z., & Marusich, L. R. (2017). Repeated measures correlation. *Frontiers in Psychology, 8*, 456. doi:10.3389/fpsyg.2017.00456

See Also

`check_dyad_residuals()`, `check_dyad_outcomes()`

Examples

```
# Data contains three compositions: female-female, female-male, and male-male.
example_data <- prepare_dyad_data(
  dyads_cross,
  dyad = coupleID,
  member = personID,
  model_types = "none",
  seed = 123
)
```

```
# This model pools all compositions and treats every dyad as exchangeable.
model <- glmmTMB::glmmTMB(
  closeness ~ 1 +
    us(1 | coupleID) +
    us(0 + .member_contrast_arbitrary | coupleID),
  dispformula = ~ 0,
  data = example_data
)

# Fewer simulations for a quick example (the default is 1000).
simulations <- simulate_dyad_responses(
  model,
  nsim = 100,
  seed = 123
)

# Check whether pooling fits the data. Each composition is checked
# separately, with role-specific summaries for distinguishable compositions.
check_partner_dependence(
  simulations,
  dyad = coupleID,
  role = gender,
  # Supply the fitting data because gender is not in the model formula.
  data = example_data
)

# Optionally, suppress plot, store object, and plot later.
check <- check_partner_dependence(
  simulations,
  dyad = coupleID,
  role = gender,
  data = example_data,
  plot = FALSE
)

plot(check, ask = FALSE, panels = TRUE)
print(check)

# For repeated occasions, also supply `member` and `time`.
example_diary_data <- prepare_dyad_data(
  dyads_ild[dyads_ild$coupleID <= 80, ], # Couples 1 to 80 are female-male.
  dyad = coupleID,
  member = personID,
  role = gender,
  time = diaryday,
  model_types = "none",
  seed = 123
)

diary_model <- glmmTMB::glmmTMB(
  closeness ~ 0 + gender + gender:diaryday +

    # In this example, we omit the correlation between partners' random
```

```
    # intercepts on level 2 with a diagonal covariance structure. This is
    # sometimes done when a model does not otherwise converge cleanly.
    diag(0 + gender | coupleID) +

    # Keep the full structure for the level 1 residuals.
    us(0 + gender | coupleID:diaryday),
  dispformula = ~ 0,
  data = example_diary_data
)

# The between level flags the omitted partner correlation.
check_partner_dependence(
  simulate_dyad_responses(diary_model, nsim = 100, seed = 123),
  dyad = coupleID,
  role = gender,
  member = personID,
  time = diaryday,
  # Supply the fitting data because personID is not in the model formula.
  data = example_diary_data
)
```

`compare_nested_models` *Compare nested glmmTMB models fitted to equivalent data*

Description

Performs a likelihood-ratio test for two nested `glmmTMB` models. The models may use ordinary data frames or dyadMLM_data objects, and their calls do not need to refer to the same R object. Models may be supplied in either order. The model with fewer estimated parameters is shown first in the result.

Usage

```
compare_nested_models(model1, model2, alpha = 0.05)
```

Arguments

`model1`, `model2`	Two fitted `glmmTMB` models to compare.
`alpha`	Significance level used for the printed conclusion. Must be one number between 0 and 1.

Details

Both model calls must use named data-frame objects that remain available when the models are compared. The checks assume these objects have not been modified since fitting. All ordinary data columns must be identical, including their types and attributes. For `dyadMLM_data`, package-generated columns may differ, but the original columns must be identical. Ordinary and prepared data may be compared with each other. Dyad metadata are checked when both models use

dyadMLM_data. The function also checks fitted rows, outcomes, weights and offsets, model family and link, maximum-likelihood estimation, and model convergence. Each model must use the same untransformed response column.

Because the test requires maximum-likelihood fits, models fitted with priors in glmmTMB::glmmTMB() are rejected. Their estimates are maximum a posteriori, not maximum likelihood.

These checks establish that the models use equivalent observations. They cannot establish that one model is mathematically nested within the other. The caller remains responsible for supplying genuinely nested models. The usual chi-squared reference distribution may also be inappropriate when tested variance parameters are on the boundary. alpha affects only the printed conclusion, not the test results.

Value

An anova-style data frame containing model degrees of freedom, information criteria, log-likelihoods, the likelihood-ratio statistic, and its chi-squared p-value. When printed, a short conclusion interprets the test at the selected significance level.

Examples

```
if (requireNamespace("glmmTMB", quietly = TRUE)) {
  restricted_data <- prepare_dyad_data(
    dyads_cross,
    dyad = coupleID,
    member = personID,
    role = gender,
    # All three observed compositions in `dyads_cross` are detected and retained
    # by default. This example focuses on `female-male` dyads, so we restrict the
    # analysis here.
    keep_compositions = "female-male",
    include_arbitrary_member_contrast = TRUE,
    seed = 123
  )
  full_data <- restricted_data

  # Compare an exchangeable model with a distinguishable model.
  restricted_model <- glmmTMB::glmmTMB(
    closeness ~ 1 +
      us(1 | coupleID) +
      us(0 + .member_contrast_arbitrary | coupleID),
    dispformula = ~ 0,
    data = restricted_data
  )

  full_model <- glmmTMB::glmmTMB(
    closeness ~ 0 + gender +
      us(0 + gender | coupleID),
    dispformula = ~ 0,
    data = full_data
  )

  # Test equal means and residual variances jointly.
```

```
  compare_nested_models(restricted_model, full_model)
}
```

---

dyads_cross *Example Gaussian cross-sectional dyadic data*

Description

A simulated long-format dataset containing distinguishable female-male dyads and exchangeable female-female and male-male dyads. Each dyad has two members and each member has one row. `closeness` and `provided_support` are the member's averages across the 14 observations in dyads_ild.

Usage

```
dyads_cross
```

Format

A data frame with 720 rows and 5 variables:

personID Unique person identifier.

coupleID Dyad identifier.

gender Gender role, with levels `female` and `male`.

closeness Mean simulated Gaussian closeness score across 14 days.

provided_support Mean simulated provided-support score across 14 days.

Details

Prepare with `dyad = coupleID`, `member = personID`, and `role = gender`. These data contain three compositions. Use `keep_compositions` in `prepare_dyad_data()` when an analysis should retain only selected compositions; omit it when all supplied compositions should remain.

Source

Simulated for `dyadMLM`; design adapted from https://github.com/Pascal-Kueng/05DyadicDataAnalysis. See Küng, P. M. (2026). *Distinguishable and Exchangeable Dyads: Bayesian Multilevel Modelling*. Zenodo. doi:10.5281/zenodo.17400655.

`dyads_ild` *Example Gaussian intensive longitudinal dyadic data*

Description

A simulated long-format dataset containing distinguishable female-male dyads and exchangeable female-female and male-male dyads. Each dyad has two members observed on 14 diary days.

Usage

```
dyads_ild
```

Format

A data frame with 10,080 rows and 6 variables:

personID Unique person identifier.

coupleID Dyad identifier.

diaryday Measurement day, from 0 through 13.

gender Gender role, with levels `female` and `male`.

closeness Simulated Gaussian closeness outcome.

provided_support Simulated provided-support score.

Details

The Gaussian outcome includes independent member-specific stationary AR(1) residual components. Their marginal standard deviation is 0.60 for both genders, with lag-1 correlations within these components of 0.55 for female members and 0.50 for male members. Separate shared and difference residuals induce same-occasion partner covariance.

Prepare with `dyad = coupleID`, `member = personID`, `role = gender`, and `time = diaryday`. These data contain three compositions. Use `keep_compositions` in `prepare_dyad_data()` when an analysis should retain only selected compositions; omit it when all supplied compositions should remain.

Source

Simulated for `dyadMLM`; design adapted from https://github.com/Pascal-Kueng/05DyadicDataAnalysis. See Küng, P. M. (2026). *Distinguishable and Exchangeable Dyads: Bayesian Multilevel Modelling*. Zenodo. doi:10.5281/zenodo.17400655.

`dyads_nbinom_cross` *Example negative-binomial cross-sectional dyadic data*

Description

A simulated long-format dataset containing the same dyads, members, and dyad compositions as dyads_cross. Each member has one row. `stress` is the member's average across the 14 observations in dyads_nbinom_ild. The count outcome is a separate negative-binomial draw using the NB2 variance function; it is not an average or sum of the daily counts.

Usage

```
dyads_nbinom_cross
```

Format

A data frame with 720 rows and 5 variables:

personID Unique person identifier.

coupleID Dyad identifier.

gender Gender role, with levels `female` and `male`.

conflict_count Simulated number of conflictual interactions reported by a member.

stress Mean simulated stress score across 14 days.

Details

Prepare with `dyad = coupleID`, `member = personID`, and `role = gender`. These data contain three compositions. Use `keep_compositions` in `prepare_dyad_data()` when an analysis should retain only selected compositions; omit it when all supplied compositions should remain.

Source

Simulated for `dyadMLM`; design adapted from `https://github.com/Pascal-Kueng/05DyadicDataAnalysis`. See Küng, P. M. (2026). *Distinguishable and Exchangeable Dyads: Bayesian Multilevel Modelling*. Zenodo. doi:10.5281/zenodo.17400655.

`dyads_nbinom_ild` *Example negative-binomial intensive longitudinal dyadic data*

Description

A simulated long-format dataset containing the same dyads, members, diary days, and dyad compositions as dyads_ild. The outcome is a count generated from a negative-binomial distribution using the NB2 variance function.

Usage

```
dyads_nbinom_ild
```

Format

A data frame with 10,080 rows and 6 variables:

personID Unique person identifier.

coupleID Dyad identifier.

diaryday Measurement day, from 0 through 13.

gender Gender role, with levels `female` and `male`.

conflict_count Simulated number of conflictual interactions reported by a member on that day.

stress Simulated stress score.

Details

Prepare with `dyad = coupleID`, `member = personID`, `role = gender`, and `time = diaryday`. These data contain three compositions. Use `keep_compositions` in `prepare_dyad_data()` when an analysis should retain only selected compositions; omit it when all supplied compositions should remain.

Source

Simulated for `dyadMLM`; design adapted from `https://github.com/Pascal-Kueng/05DyadicDataAnalysis`. See Küng, P. M. (2026). *Distinguishable and Exchangeable Dyads: Bayesian Multilevel Modelling*. Zenodo. doi:10.5281/zenodo.17400655.

prepare_dyad_data *Prepare dyadic data for multilevel models*

Description

Validates dyadic data, records the structural variables, and adds metadata and model-ready columns for dyadic multilevel model parameterizations.

Usage

```
prepare_dyad_data(
  data,
  dyad,
  member,
  role = NULL,
  time = NULL,
  predictors = NULL,
  lag1_predictors = NULL,
  model_types = "apim",
  dsm_role_order = NULL,
  temporal_decomposition = c("auto", "2l", "none"),
  set_exchangeable_compositions = NULL,
  keep_compositions = NULL,
  pool_compositions = NULL,
  incomplete_dyads = c("error", "drop"),
  missing_role = c("error", "drop"),
  seed = NULL,
  short_colnames = TRUE,
  include_arbitrary_member_contrast = FALSE,
  add_apim_gmc_predictors = FALSE
)
```

Arguments

data	A data frame or tibble. Data must be in long format. For cross-sectional dyadic data, each observed member of each dyad has one row. For intensive longitudinal dyadic data, each observed member of each dyad has one row per observed time point.
dyad	Column identifying the dyad.
member	Column identifying a person or the member within dyad.
role	Optional column identifying a stable member role, such as gender. Non-missing values must be consistent within each dyad x member and must not contain _x_. In repeated-measures data, an observed role is propagated to missing rows for the same member within a dyad. missing_role controls dyads in which a member has no non-missing role on any row. If no role is supplied, all dyads are treated as the same type of exchangeable dyads.

`time` Optional column identifying time or measurement order of repeated measures.

`predictors` Optional variables to use for temporal predictor decomposition and model-ready predictor construction.

`lag1_predictors`

Optional subset of `predictors` for which lag-1 model-ready columns should be created. Requires `time` to be a finite, integer-valued numeric measurement index. Lagging respects the dyad and member structure, matches observations at exactly `time - 1`, and does not bridge missing occasions. If occasions are unequally spaced, for example waves in 2008, 2010, and 2013, number the study's occasions (1, 2, 3) and supply that index as `time`. The lag then refers to the previous occasion, regardless of how much time lies between them. Eligible raw, within-person, and APIM GMC variables are lagged, but stable between-person variables are not.

`model_types` Model-ready column families to construct. Can contain one or more of `"apim"`, `"dim"`, and `"dsm"`. `"apim"` creates actor and partner predictors. `"dim"` creates dyad-mean and within-dyad member-deviation predictors. DIM requires one final exchangeable dyad composition, which may result from pooling exchangeable compositions. `"dsm"` creates dyadic-score model predictor columns. `"none"` skips model-specific predictor construction after validation, composition inference, and optional temporal predictor decomposition, and must be used alone. `"dim"` and `"dsm"` must be requested in separate calls.

`dsm_role_order` For `model_types = "dsm"`, a character vector giving the two distinguishable roles in the order used for directional differences. For example, `c("female", "male")` defines predictor differences as female minus male and assigns the DSM role contrast `+0.5` to female partners and `-0.5` to male partners. Required when DSM columns are requested and must be `NULL` otherwise.

`temporal_decomposition`

Temporal decomposition strategy for `predictors`. `"none"` leaves predictors undecomposed before model-specific columns are constructed. `"2l"` indicates a two-level temporal predictor decomposition into within-person and between-person components. `"auto"` resolves to `"2l"` when both `time` and `predictors` are supplied, and to `"none"` otherwise. `"2l"` retains raw model-ready predictors in addition to their within-person and between-person components. For longitudinal DIM and DSM construction, raw and within-person dyadic scores are computed within each dyad occasion, while between-person scores are computed within dyads. Raw DIM and DSM dyad means are grand-mean centered. Do not include the raw, within-person, and between-person versions of the same contemporaneous predictor in one model because they are linearly dependent.

`set_exchangeable_compositions`

Optionally specify dyad compositions to treat as exchangeable, when their roles would otherwise imply distinguishability. Requires `role`. Compositions that are already exchangeable should not be listed. Each composition must be supplied as one string, using `_x_`, `-`, `_`, or whitespace ( ) between the two role labels, for example `"female_x_male"`, `"female-male"`, `"female_male"`, or `"female male"`, in arbitrary order. To set multiple compositions, use a character vector of such strings. Unlike `include_arbitrary_member_contrast`, this reclassifies the selected compositions as exchangeable and changes their composition-role

coding and indicators, which is required for composition pooling and DIM compatibility.

`keep_compositions`
: Optional observed dyad compositions to keep before exchangeability overrides and pooling. Requires `role`. Composition references use the same format as `set_exchangeable_compositions`. `NULL` keeps all observed compositions.

`pool_compositions`
: Optionally pool exchangeable dyad compositions into a shared final composition label. Must be a named list where each name is the final composition label and each value is a character vector of composition references, for example `list(same_sex_couples = c("female-female", "male-male"))`. Only exchangeable compositions can be pooled. Each pool must contain at least two distinct observed compositions after composition references are resolved.

`incomplete_dyads`
: How to handle dyads with fewer than two unique members across all rows in `data`. `"error"` stops with an error and `"drop"` removes the entire dyad. A dyad with more than two unique members is invalid and always causes an error, regardless of this setting.

`missing_role`
: How to handle dyads in which at least one member has no non-missing `role` value on any row. A consistent non-missing role observed for a member is propagated to that member's other rows before this policy is applied. `"error"` stops with an error and `"drop"` removes the entire dyad. Conflicting non-missing roles always cause an error. Ignored when no `role` column is supplied.

`seed`
: Optional seed for random `.member_contrast_*` sign assignment. If `NULL`, the current R session's RNG state is used.

`short_colnames`
: Whether to use shorter composition-dependent generated column names when the final data contain one composition. The default `TRUE` omits the redundant composition label from `.is_*` and `.member_contrast_*` names. `FALSE` always retains composition-qualified names. Other generated column names are unaffected.

`include_arbitrary_member_contrast`
: Whether to also generate arbitrary `.member_contrast_*` columns for distinguishable compositions. The default `FALSE` generates these columns only for exchangeable compositions. `TRUE` retains distinguishable composition metadata, composition roles, and role indicators. It only adds the contrast needed to fit an exchangeability-constrained random-effects model to the same prepared data. Use `set_exchangeable_compositions` instead to reclassify compositions for pooling or DIM compatibility.

`add_apim_gmc_predictors`
: Whether to add APIM GMC variants for numeric predictors. `TRUE` retains the raw columns and adds `.{pred}_gmc` plus its actor and partner versions, centered over all retained non-missing values after filtering. Lagged variants use the same mean. Mixed non-numeric predictors remain raw and are listed in one warning. In longitudinal data, the mean is observation-weighted. Requires `"apim"` in `model_types`, at least one numeric predictor, and resolved `temporal_decomposition = "none"`. Do not use raw and GMC variants together in a model with an intercept.

Details

Data must be in long format. Cross-sectional dyadic data may contain at most one row per member within dyad. Intensive longitudinal dyadic data may contain at most one row per member and observed measurement occasion within dyad. Measured variables may contain missing values. Numeric `dyad`, `member`, `time`, and selected predictor columns must not contain infinite values. Structural completeness is assessed across all rows. `incomplete_dyads` controls dyads with fewer than two members; dyads with more than two members always cause an error. When `role` is supplied, stable member roles are resolved across repeated rows before `missing_role` is applied.

Dyad composition labels are canonical: role labels are sorted alphabetically before being combined, so labels do not depend on row or member order.

Value

The original data as a tibble with class `dyadMLM_data`, `.composition` and `.composition_role` factor columns, `.is_*` numeric indicator columns, and numeric `.member_contrast_*` columns coded `-1` and `1` for the two members of each matching composition and `0` otherwise. With one final composition, their default names omit the composition label. The `dyadMLM` attribute contains structural metadata, `dyad_compositions`, and predictor metadata such as `temporal_decompositions`, `lag1_predictors`, `apim_predictors`, and `dim_predictors`, as well as `dsm_predictors` and `dsm_role_order` when applicable. The `generated_columns` table records each package-generated column retained in the returned data. Modifying prepared data does not automatically regenerate package-generated columns.

Examples

```
data <- data.frame(
  dyad_id = c(1, 1, 2, 2, 3, 3),
  person_id = c(1, 2, 3, 4, 5, 6),
  role = c("female", "male", "female", "female", "male", "male"),
  x = c(4, 7, 5, 6, 3, 8)
)

prepared <- prepare_dyad_data(
  data,
  dyad = dyad_id,
  member = person_id,
  role = role,
  predictors = x,
  model_types = "apim"
)

print(prepared)

pooled <- prepare_dyad_data(
  data,
  dyad = dyad_id,
  member = person_id,
  role = role,
  predictors = x,
  model_types = "apim",
```

```
  set_exchangeable_compositions = "female-male",
  pool_compositions = list(
    romantic_couples = c("female-female", "male-male", "female-male")
  )
)

print(pooled)

ild_data <- data.frame(
  dyad_id = rep(c(1, 2), each = 4),
  person_id = rep(c(1, 2), times = 4),
  time = rep(c(1, 1, 2, 2), times = 2),
  x = c(4, 7, 5, 8, 3, 6, 4, 7)
)

ild_prepared <- prepare_dyad_data(
  ild_data,
  dyad = dyad_id,
  member = person_id,
  time = time,
  predictors = x,
  lag1_predictors = x,
  model_types = "apim",
  seed = 123
)

print(ild_prepared)
```

`recover_exchangeable_covariance` *Recover member-level residual covariance from exchangeable random-effect blocks*

Description

Back-transforms covariance matrices from paired shared and member-difference random-effect blocks to the covariance structure of two exchangeable members. The result is on the fitted random effects' linear-predictor scale. In non-Gaussian models, it therefore describes a Gaussian latent covariance, not response-scale residual covariance. For the model specification, derivation, and interpretation, see the exchangeable APIM vignette.

Usage

```
recover_exchangeable_covariance(
  model,
  block_pairings = NULL,
  posterior = c("mean", "median", "draws")
)
```

Arguments

model	A fitted glmmTMB or single-response brmsfit model. For glmmTMB, only conditional-model random effects are processed.
block_pairings	NULL (default) for automatic block matching. Otherwise, supply one block-pair specification or a list of block-pair specifications. Each pair contains: • shared_block: a single string naming the shared random-effect term copied from the fitted model formula or an equivalent selector (see Details), or NULL if the entire shared block was omitted when fitting; • difference_block: a single string naming the member-difference random-effect term or an equivalent selector, or NULL if the entire difference block was omitted; • difference_indicator: the exact name of the difference-indicator column used in difference_block. It is required when difference_block selects a block and optional when difference_block = NULL; • shared_indicator: the exact shared composition-indicator column, coded 1 on the rows of this composition and 0 elsewhere, needed only for composition-specific blocks in mixed-dyad models. It defaults to "1", meaning that every fitted row belongs to the pair and an ordinary intercept is the shared intercept coordinate.
posterior	For brms models, return posterior "mean" matrices (default), "median" matrices, or full posterior "draws". Ignored for glmmTMB models.

Details

Automatic matching recognizes the compact .member_contrast_arbitrary and composition-qualified .member_contrast_*_arbitrary names. It first looks for the corresponding .is_* shared block. It requires the two blocks to use the same grouping factor and the same underlying terms. Most models fitted with dyadMLM-generated columns therefore need only:

```
result <- dyadMLM::recover_exchangeable_covariance(model)
print(result)
```

Supply block_pairings when automatic matching is ambiguous or when a model uses custom indicators or deliberately omitted blocks or terms. To specify one pair with a custom difference indicator:

```
result <- dyadMLM::recover_exchangeable_covariance(
  model,
  block_pairings = list(
    shared_block = "(1 + time | coupleID)",
    difference_block = "(0 + my_diff + I(my_diff * time) | coupleID)",
    difference_indicator = "my_diff"
  )
)
```

For multiple covariance levels, wrap the pairings in an outer list. For example, in a Gaussian glmmTMB model fitted with dispformula = ~ 0, this call recovers both a stable dyad-level covariance with an omitted difference time slope and the same-occasion partner residual covariance:

```
result <- dyadMLM::recover_exchangeable_covariance(
  model,
  block_pairings = list(
    dyad = list(
      shared_block = "(1 + diaryday | coupleID)",
      difference_block = "(0 + .member_contrast_arbitrary | coupleID)",
      difference_indicator =
        ".member_contrast_arbitrary"
    ),
    same_occasion = list(
      shared_block = "(1 | coupleID:diaryday)",
     difference_block = "(0 + .member_contrast_arbitrary | coupleID:diaryday)",
      difference_indicator =
        ".member_contrast_arbitrary"
    )
  )
)
```

At the dyad level, the fitted model includes a shared time slope but no difference time slope. Thus, the two members' time random effects are identical at this level, with correlation +1 whenever the shared slope variance is non-zero; at zero variance, the correlation is undefined. Covariances involving the diary-day slope are therefore supplied entirely by the shared block.

The random-effect terms may be copied exactly from the model formula. Equivalent backend syntax is also recognized, such as `(1 + time | group)` and `us(1 + time | group)`, or `(0 + x || group)` and `diag(0 + x | group)`. A custom difference-indicator name is supplied literally. Difference slopes may be written in either interaction order or as a simple product inside `I()`.

When a difference block is supplied and the fitted model frame retains the indicator columns, `difference_indicator` must assign `-1` and `+1` to the two arbitrary member positions consistently within each dyad. For composition-specific blocks, it must be zero where `shared_indicator` is zero.

The function checks whether both positions occur within each supported fitted grouping unit. It rejects coding when no group contains both positions. For custom difference indicators supplied through `block_pairings`, it also warns when only some groups are one-sided, which can result from fitted-row filtering, and when stable member assignments cannot be verified across repeated rows without a member identifier.

Value

An `exchangeable_covariance` object: a named list with one element per matched block pairing. Each element contains `shared_term` and `difference_term` (each a fitted term string or `NULL`), plus the member-level variance-covariance matrix in `varcov` and its standard-deviation/correlation representation in `sdcor`, with standard deviations on the diagonal and correlations off the diagonal. Covariance dimension labels `member_1` and `member_2` denote arbitrary exchangeable positions, not substantive roles or encodings. Caller-supplied outer names are preserved; unnamed pairings receive stable names `pair_1`, `pair_2`, and so on in resolved order. `varcov` and `sdcor` are matrices, except when `posterior = "draws"` for a `brms` model, where they are draw by coefficient by coefficient arrays. `brms` quantities are transformed draw by draw before summarizing.

What omitted blocks and terms mean

`recover_exchangeable_covariance()` only describes constraints that were already imposed when the model was fitted. It does not remove a block, set a variance to zero, or otherwise constrain the supplied model. Describe only the structure that was actually fitted.

If a term occurs in only one selected block, the function represents the missing coordinate as a structural zero:

- A term present only in the shared block has no difference component, so the two members have identical random effects for that term.
- A term present only in the difference block has no shared component, so the two members have equal-magnitude, opposite-sign random effects for that term.

Setting `difference_block = NULL` or `shared_block = NULL` applies the corresponding rule to the entire omitted block. This is valid only when that block is truly absent from the fitted model. Do not use `NULL` merely to ignore an existing block; the resulting back-transformation would be incorrect.

See the constrained-block example in the exchangeable APIM vignette.

Backend note

For `glmmTMB`, random effects in non-conditional components are ignored with a warning. See the vignette's extension to exchangeable random slopes section for the manual calculation.

In `brms`, cross-sectional and same-occasion partner dependence can be represented directly with `unstr(time = member_position, gr = residual_group)`. With Gaussian outcomes, `sigma ~ 1` supplies the common residual scale. For families such as Poisson and negative binomial, `unstr()` adds a latent residual scale and correlation on the linear-predictor scale. Here, `member_position` identifies the same two arbitrary positions within every group, and `residual_group` identifies dyads in cross-sectional data or dyad-occasions in longitudinal data. This direct specification applies when one covariance structure is sufficient. Separate composition-specific `unstr()` structures for mixed dyad types are not currently supported in a standard single-response `brms` model. For Gaussian mixed-dyad residual covariance, use `glmmTMB`. Shared/difference blocks remain relevant for higher-level random effects and can represent latent link-scale covariance in non-Gaussian models.

See Also

The exchangeable APIM vignette for the model specification, covariance derivation, and constrained-block example. Run `vignette("apim", package = "dyadMLM")` to open the installed version.

Examples

```
if (requireNamespace("glmmTMB", quietly = TRUE)) {
  example_data <- prepare_dyad_data(
    dyads_cross,
    dyad = coupleID,
    member = personID,
    role = gender,
    model_types = "none",
    # All three observed compositions in `dyads_cross` are detected and retained
    # by default. This example focuses on `female-female` dyads, so we restrict
```

```
  # the analysis here.
  keep_compositions = "female-female",
  seed = 123
)

model <- glmmTMB::glmmTMB(
  closeness ~ 1 +
    us(1 | coupleID) +
    us(0 + .member_contrast_arbitrary | coupleID),
  dispformula = ~ 0,
  data = example_data
)

recover_exchangeable_covariance(model)
}
```

`simulate_dyad_responses` *Simulate response datasets for predictive checks*

Description

[Experimental] If a model fits, data simulated from it should resemble the observed data. This function generates new response datasets for the same observations and predictors. Reuse them with `check_partner_dependence()`, `check_dyad_residuals()`, or `check_dyad_outcomes()` to check whether a fitted model reproduces features of the observed data. For a complete example see `check_partner_dependence()`.

Usage

```
simulate_dyad_responses(model, nsim = 1000, seed = NULL)
```

Arguments

`model`	A fitted `glmmTMB` model.
`nsim`	Number of complete response datasets to simulate. Default: 1000.
`seed`	Optional seed for reproducible simulations, interpreted as an integer (see `set.seed()`). When supplied, the random-number state is restored after the function returns, or when it stops after an error.

Value

A `dyadMLM_response_simulations` object for use with `check_partner_dependence()`, `check_dyad_residuals()`, and `check_dyad_outcomes()`.

The result keeps all components in fitted-row order (after missing-data exclusions):

- `simulated_responses`: a matrix with `nsim` rows and one column per fitted observation. Each row is a complete simulated dataset.
- `observed_response` and `predicted_response`: numeric vectors with one value per fitted observation.
- `model_frame`: the fitted model frame (variables in the model formulas only), in the same row order.

The `dyadMLM` attribute records the model and simulation settings, including the seed.

Supported models

Supports unweighted `glmmTMB` models with the following families:

- `gaussian()`
- `poisson()`, `glmmTMB::compois()`, `glmmTMB::genpois()`, `glmmTMB::bell()`
- `glmmTMB::nbinom1()`, `glmmTMB::nbinom2()`, `glmmTMB::nbinom12()`
- `glmmTMB::truncated_poisson()`, `glmmTMB::truncated_nbinom1()`, `glmmTMB::truncated_nbinom2()`, `glmmTMB::truncated_compois()`, `glmmTMB::truncated_genpois()`
- `glmmTMB::tweedie()`
- `Gamma()`, `glmmTMB::ziGamma()`
- `glmmTMB::beta_family()`
- `glmmTMB::lognormal()`
- `glmmTMB::skewnormal()`
- `glmmTMB::t_family()` with more than two degrees of freedom
- `glmmTMB::ordinal()` (glmmTMB 1.1.15.2 or later)

Smooth terms (`s()`) are not supported: simulations would draw new curves. Fixed-effect splines such as `splines::ns()` are supported.

Zero-inflated and hurdle versions are supported where available. Checks describe the combined response, including zeros, rather than each model component separately. Good agreement does not establish that the zero and response components each fit well.

Ordinal checks use category scores `1, 2, ..., K` in their fitted order, matching glmmTMB's predictions. The checks compare observed and simulated scores. The scores do not measure distances on an underlying continuous scale. Because each dyad contains only two people, glmmTMB tends to underestimate the dyad variance in ordinal models. Simulations then show less partner dependence than the data, so the partner check can flag a correct ordinal model, especially with many dyads.

The model's fitted link is used for prediction and simulation. Predictions and simulated responses must be finite.

Technical details

Each simulation draws new random effects, if the model has any, and then new responses from the fitted model. Random effects within each block are drawn together using their fitted variances and correlations. This also applies to random effects in the zero-inflation and dispersion models, if present.

Fitted parameters and predictors, including any lagged responses, stay fixed. The model is not refitted, and uncertainty in parameter estimates is not included. This is a *plug-in predictive reference* (Gelman et al., 1996). If all grouping factors are dyads or nested within dyads (such as dyad-days), the simulations represent hypothetical new dyads under the same study design.

`predicted_response` contains predicted mean responses with random effects in the conditional and zero-inflation models set to zero. For zero-inflated and hurdle models, this is the conditional response mean multiplied by one minus the zero-component probability (Brooks et al., 2017, Appendix A; doi:10.32614/RJ2017066).

Partner-dependence checks (by default) and the response SD and largest absolute deviation of `check_dyad_outcomes()` subtract these same predictions from observed and simulated responses. Both random effects and observation-level noise still contribute to response variance. With nonlinear links, setting random effects to zero generally differs from averaging predictions over them.

References

Gelman, A., Meng, X.-L., & Stern, H. S. (1996). Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica, 6*(4), 733–807.

Index

∗ **datasets**
dyads_cross, 15
dyads_ild, 16
dyads_nbinom_cross, 17
dyads_nbinom_ild, 18

check_dyad_outcomes, 3
check_dyad_outcomes(), *5*, *7*, *11*, *27*, *29*
check_dyad_residuals, 5
check_dyad_residuals(), *3, 4*, *11*, *27*
check_partner_dependence, 8
check_partner_dependence(), *4*, *6, 7*, *27*
compare_nested_models, 13
compare_nested_models(), *10*

dyadMLM_data, *13*
dyads_cross, 15, *17*
dyads_ild, *15*, 16, *18*
dyads_nbinom_cross, 17
dyads_nbinom_ild, *17*, 18

glmmTMB's predictions, *28*

plot.dyadMLM_outcome_check(), *3*
plot.dyadMLM_residual_check(), *5*
prepare_dyad_data, 19
prepare_dyad_data(), *15–18*

recover_exchangeable_covariance, 23

set.seed(), *27*
simulate_dyad_responses, 27
simulate_dyad_responses(), *3–6*, *9*